在 DOS 新版本下运行王码、金山及 2.13H 汉字系统

宋运康 （重庆市大足师范）

摘要：文章分析了在 *MS DOS 5.0* 和 *DRDOS 6.0* 下运用流行汉字系统不能完全、充分发挥 *MS DOS 5.0* 和 *DR DOS 6.0* 性能的原因，给出了具体的实施方案，较好地解决了国内广大用户普遍关心的问题。

无论是国内或国外，MS DOS 5.0 和 DR DOS 6.0 都是目前在单用户个人计算机上运行的最优秀的 DOS 操作系统。它们在继承以往的 DOS 操作系统的基础上，增加或增强了许多新的功能，特别是 DR DOS 6.0，它使许多用户爱不释手。对于国内用户，大家最关心的问题显然是在新的操作系统下面能否使用流行的汉字系统。

这是因为：无论是 MS DOS 5.0 或 DR DOS 6.0，它较之旧版本的强大功能是以增大系统驻留文件为条件的。单就三个必备的系统驻留文件而言，MS DOS 5.0 比 DOS 3.0 增大了约 20k，DR DOS 6.0 比 3.30 增大了约 40k，超过 117k。

另一方面，在 MS DOS 5.0 或 DR DOS 6.0 的众多优秀功能中，首推对于扩展内存的利用，它们把庞大的多个系统驻留文件移往 640K 常规内存之外，从而使用户较低版本 DOS 有更大的自由空间。但这个最主要的优点却与汉字系统的庞大字库相冲突，众所周之，容纳国标二级字库至少需要近 240K 空间。

造成 MS DOS 5.0 或 DR DOS 6.0 的驻留文件与汉字库不能共处于扩展内存的矛盾，是因为两者都抢占扩展内存的下部（最靠近 640K 常规内存的部分）。

本文旨在解决 MS DOS 5.0 或 DR DOS 6.0 的系统驻留文件与汉字库同时享用 640K 之外的扩展空间的矛盾，在和以往低版本 DOS 完全一样运行汉字系统的同时，完整、充分地发挥 MS DOS 5.0 或 DR DOS 6.0 的各种功能。

解决的途径是要么修改 MS DOS 5.0 或 DR DOS 6.0 系统，要么修改汉字系统的字库装载和字模查找部分。显然修改后者的难度要低得多，且不会有任何后遗症。修改思路是：启动 MS DOS 5.0 或 DR DOS 6.0 的驻留文件进入扩展内存之后，强迫汉字库装入扩展内存中不与 MS DOS 5.0 或 DR DOS 6.0 的文件相重叠的位置，并修改汉字字模查找部分，使其能在新的位置中找到。确定新位置的起始处的最简方法是：先找到原字库装载起始位置，再在此基础上根据机器内存大小和需要向上方平移。笔者推荐把字库移到扩展内存的最上方，此时，可按如下关系计算：

平移距离＝机器内存总容量+384K−字库总容量−原字库安装位置起始地址

其中的机器内存总容量按一般说法，如对于一般 286 机是 1M，这里的 384K 是系统为显示缓冲区等用途保留。

这里，需要简单介绍一下上述三个汉字系统是如何把字库装入扩展内存和如何在扩展内存查找所需汉字字模的。

2.13H 把扩展空间设置成虚拟盘，再把字库拷入，字库的使用同其它非虚拟盘上类似。它的缺点是需要虚拟盘管理程序。

王码 5.0 和金山 5.0 都是利用 80286 以上机的 BIOS 中断 INT15 的 87、88 号实现汉字库在扩展内存的存取。88 号功能检测机器的扩展内存数量，入口参数是 AH＝87H，出口参数是 AX＝扩展内存数量（单位是 K 字节）。87 号功能是在 16M 内存空间内（包含常规内存的 640K）传递内存内容，调用前需要选设置一个称作全局描述符表（GDT）的参数块，该参数块由 6 个描述符组成，每个描述符为 8 个字节，结构如下：

00H　空描述字
08H　全局描述符表描述字
10H　源地址描述字
18H　目的地址描述字
20H　ROM BIOS 代码描述字
28H　ROM BIOS 栈描述字

与本文有密切关系的是第三、第四项描述符(即源地址描述符和目的地址描述符)的 3-5 字节(其他内容请参见有关资料),它们分别指定要移动的内存数据块的源地址和目的地址,要注意的是它们都是 24 位的线性地址(范围是 000000H-FFFFFFH),不是通常的段:偏移量的形式。但仍是低位在前,高位在后。

再设置入口参数:

AH=87H

CX=传递数量(单位是字)

ES:SI=参数块首址

出口参数为:调用成功:ZF 为 1,AH 为 0;

调用失败:ZF 为 0,AH 为错误码。

以下介绍用 DEBUG 对上述三个汉字系统的具体修改方法。并对有关片断作尽可能简略的注释。

1.金山 5.0

```
debug chlib.com
-s100 51do cd 15;查找 int 15
116C: 4A25
116C: 4D5F
116C:4E16
-u4e16 4e1b
116C: 4E14 B488      MOV   AH,88
116C: 4E16 CD15      INT   15
116C: 4E18 3DEE00    CMP   AX,00EE;检测是否至少有240K扩展内存
116C: 4E1B 7202      JB    4E1F   ;若不足,跳转
-u49f4 4a25   ;此段为从扩展内存取用所需字模
116C: 49F4 81E27F7F  AND   DX,7F7F; 屏蔽所取汉字内码的最高位
116C: 49F8 81EA2121  SUB   DX2121; 计算该汉字的区位码(区号在 DH,位号在 DL)
116C: 49FC 80FE09    CMP   DH,09 ; 区号大于 9?
116C: 49FF 7203      JB    4A04; 否,跳转
116C: 4A01 80EE06    SUB   DH,06; 对大于 9 的区号减 6,跳过 10-15 空区
116C: 4A04 8BC2      MOV   AX,DX
116C: 4A06 B05E      MOV   AL,5E
116C: 4A08 F6E4      MUL   AH; 区号乘以 94(5EH)
116C: 4A0A 32F6      XOR   DH,DH
116C: 4A0C 03C2      ADD   AX,DX; 再加上位号;
116C: 4A0E BA2000    MOV   DX,0020
116C: 4A11 F7E2      MUL   DX; 乘以 32(20H),得到该汉字相对于字库头的位移量
116C: 4A13 A31949    MOV   [4919], AX;此位移量低 16 位送 GDT 源地址描述符相应位置
116C: 4A16 80C210    ADD   DL,10 ; 此位移量高 8 位加 10H
116C: 4A19 88161B49  MOV   [491B],DL;此位移量高 8 位送 GDT 源地址描述符相应位置
116C: 4A1D B91000    MOV   CX,0010;欲取 16 个字(32 字节,一个汉字的字模)
116C: 4A20 BE0749    MOV   SI,4907 ; GDT 首址送 ES:SI
116C: 4A23 B487      MOV   AH,87
116C: 4A25 CD15      INT   15
-u4d3b 4d79          ;此段为装入字库到扩展内存。
116C: 4D3B 56        PUSH  SI ; SI 值为汉字库未装部分的容量,单位是节(16 字节),第一次调用前为全字库容量 3B7CH
116C:4D3C B90008     MOV   CX,0800; 调用一次欲装入 800H 节
116C:4D3F 81FE0008   CMP   SI,0800; 未装容量是否少于 800H 节?
116C:4D43 7302       JNB   4D47 ;不少于,跳转
116C:4D45 8BCE       MOV   DX,SI; 少于,则使欲装容量为剩余容量
116C:4D47 BAA851     MOV   DX,51A8; 取装入缓冲区首址,(在常规内存)
116C:4D4A D1E1       SHL   CX,14
116C:4D4C D1E1       SHL   CX,1
116C:4D4E D1E1       SHL   CX,1
116C:4D50 D1E1       SHL   CS,1; 把该次欲装入容量换算成字节
116C: 4D51           MOV   AH,3F
116C:4D54 CD21       INT   21; 读磁盘字库文件
116C:4D56 BE0749     MOV   SI,4907; 取 GDT 首址
116C:4D59 8BC8       MOV   CX,AX; 传送字节数送 CX
116C:4D5B D1E9       SHR   CX,1 ; 换算成字数
116C:4D5D B487       MOV   AH,87
116C:4D5F CD15       INT   15; 传送到扩展内存
116C:4D61 810621490080 ADD WORD PTR [4921],8000; 修改传送目的地址低 16 位
```

```
116C:4D67 8016234900    ADC   BYTE PTR [4923],00; 修改
                                目的地址高 8 位
116C:4D6C 5E            POP   SI
116C:4D6D 81EE0008      SUB   SI,0800; 修改字库未装入量
116C:4D71 7204          JB    4D77 ; 若小于 0,跳转 4D77
116C:4D73 7402          JZ    4D77; 若等于 0,跳转 4D77
116C:4D75 EBC4          JMP   4D3B; 若大于 0,跳转 4D3B,
                                继续装
116C:4D77 B43E          MOV   AH,3E
116C:4D79 CD21          INT   21; 关闭磁盘字库文件
```

当跟踪到 116C:4d5d 处,查看原程序字库将装入扩展内存的起始位置:

```
-d4907 492f
116C:4900                    00-00 00 00 00 00 00 00 00
116C:4910 00 00 00 00 00 00 00 00-80 A8 51 00 93 00 00 00
116C:4902 80 00 00 10 93 00 00 00-00 00 00 00 00 00 00 00
```

知原起始位置是 100000H.此时可根据前述公工算得字库在扩展内存的新地址较旧地址的位移量,若机器内存为 1M,则:

位移量=1M+384K-3b7c0H-100000H=024840H

现在可作如下修改:

-A51d1 51e0 ; 下列语句附加到 chlib.com 尾部,为修改 GDT 目的描述符中的数据传送目的地址初值,它将在程序驻留时和原程序的尾部一起被截掉,不致占用内存。

```
117D:51D1 53          PUSH  BX
117D:51D2 BB4048      MOV   BX,4840
117D:51D5 011E2149    ADD   [4921],BX
117D:51D9 B302        MOV   BL,02
117D:51DB 101E2349    ADC   [4923],BL
117D:51DF 5B          POP   BX
117D:51E0 C3          RET
```

-A2f30 2f42 ; 下列子程序为修改 GDT 源描述符中的数据传送源地址,将被多次调用,不能附加到原程序尾,笔者把它放入原程序中部绝对安全的数据区,无副作用。

```
117D:2F30 53          PUSH  BX
117D:2F31 BB4048      MOV   BX,4840
117D:2F34 011E1949    ADD   [4919],BX
117D:2F38 B302        MOV   BL,02
117D:2F3A 101E1B49    ADC   [491B],BL
117D:2F3E BE0749      MOV   SI,4907
117D:2F41 5B          POP   BX
117D:2F42 C3          RET
-A100
117d:100 jmp 51d1
117d:103 jmp 4c30
```

最后用 R 命令修改 cx 中的文件长度为 50E1,用 W 命令存盘。

2.王码 5.0

```
debug w5sk.wm5
-s100 15fe cd 15
116C:0E91
116C:112F
116C:1173
116C:140a
-u0e8f 0e96     ;此段检测扩展内存容量
116C:0E8F B488      MOV   AH,88
116C:0E91 CD15      INT   15
116C:0E96 3D2C01    CMP   AX,012C ;判断是否至少有 300K
                                扩展内存
116C:0E96 722C      JB    0EC4 ; 若不足,跳转
-u1123 112f   ;此段装入字库,每调用一次从常规内存中传送 32K
                                到扩展内存
116C:1123 E8FPFE    CALL  1016 ; 涉及语句太多,不便注释
116C:1126 8D36E401  LEA   SI, [01E4] ;从 ds:01e4 单元取出
                                GDT 首址
116C:112A B487      MOV   AH,87
116C:112C B90040    MOV   CX,4000 ; 每次传送 32K 字节
116C:112F CD15      INT   15
-u1166 1173   ;此段的作用仍为装入字库,这是 WM5.0 的特殊
116C:1166 E8ADFE    CALL  1016
116C:1169 59        POP   CX
116C:116A 41        INC   CX
116C:116B D1E9      SHR   CX,1
116C:116D 8D36E401  LEA   SI,[01E4]
116C:1171 B487      MOV   AH,87
116C:1173 CD15      INT   15
-u1400 140a   ;从扩展内存中取用所需要字模
116C:1400 07        POP   ES
116C:1401 8D36E401  LEA   SI,[01E4]
116C:1405 B487      MOV   AH,87
```

```
116C:1407 B90002      MOV   CX,0200 ; 取用 512 字节,而不是
                            通常的 32 字节,是系统的又一特殊
116C:140A CD15        INT   15
```

首先修改 116c:0e96 处,以强迫字库装入新位置:

```
-a116c:0e96
116c:0e96 nop
116c:0e97 nop
```

再修改下列三处,以调用新增的子程序,使字库的装入、取用都到新的地址:

```
-a1126 1130
116C:1126 B90040      MOV   CX,4000
116C:1129 E86600      CALL  1192
116C:112C 90          NOP
116C:112D 90          NOP
116C:112E 90          NOP
116C:112F 90          NOP
116C:1130 90          NOP
-a116d 1174
116C:116D B82200      CALL   1192
116C:1170  90         NOP
116C:1171 90          NOP
116C:1172 90          NOP
116C:1173 90          NOP
116C:1174 90          NOP
-a1401 140b
116C:1401 B90002      MOV    CX,0200
116C:1404 E8ACFD      CALL   11B3
116C:1407 90          NOP
116C:1408 90          NOP
116C:1409 90          NOP
116C:140A 90          NOP
116C:140B 90          NOP
```

下列子程序段使系统把字库装入新位置,由于此子程序将被连续几次调用(调用一次装入只能 32K),而原程序又自动修改每次调用时的目的地址(叠加 32K),所以下列子程序段在 int 15 之后再使目的地址还原,以抵消原程序中的叠加。(也可修改原程序目的地址叠加部分,但较繁)。

再压缩位于 118a-11c8 处的一句较长的提示"Non library .....¥",为新增语句挤出安全空间:-e118a "no lib!¥" 然后增写如下程序段:

```
-a1192 11C8
116C:1192 8D36E401    LEA   SI,[01E4]
116C:1196 B84048      MOV   AX,4840
116C:1199 26          ES:
116C:119A 01441A      ADD   [SI+1A],AX ;修改 GDT 目的地
                            址低 16 位
116C:119D B002        MOV   AL,02
116C:119F 26          ES:
116C:11A0 10441C      ADC   [SI+1C],AL ; 修改 GDT 目的地
                            址高 8 位
116C:11A3 B487        MOV   AH,87
116C:11A5 CD15        INT   15
116C:11A7 B84048      MOV   AX,4840
116C:11AA 29441A      SUB   [SI+1A],AX ; 使 GDT 目的低 16
                            位还原
116C:11AD B002        MOV   AL,02
116C:11AF 28441C      SUB   [SI+1C],AL ;使 GDT 目的高 8 位
                            还原
116C:11B2 C3          RET
```

下列子程序使系统在新的位置取得汉字点阵:

```
-a11b3
116C:11B3 8D36E401    LEA   SI,[01E4]
116C:11B7 B84048      MOV   AX,4840
116C:11BA 26          ES:
116C:11BB 014412      ADD   [SI+12],AX ;修改 GDT 源地址
                            低 16 位
116C:11BE B002        MOV   AL,02
116C:11CO 26          ES:
116C:11C1 104414      ADC   [SI+14],AL;修改 GDT 源地址
                            高 8 位
116C:11C4 B487        MOV   AH,87
116C:11C6 CD15        INT   15
116C:11C8 C3          RET
```

3.2.13 系统

由于该系统是采用较落后的虚拟盘方式实现字库在扩展内存的存取,需要重新编写一段采用 INT 15 的字库装入程序(不驻留),并修改或重新编制 INT 7F 中断驻留程序(读显示字模中断)。考虑到篇幅,不再给出具体程序。欢迎来信索取。